Exhibit 3

Power management

Android 9 (API level 28) introduces new features to improve device power management. These changes, along with features that were already present in previous versions, help to ensure that system resources are given to the apps that need them the most.

The power-management features fall into two categories:

App standby buckets (#buckets)

The system limits apps' access to device resources like the CPU or battery, based on the user's usage patterns. This is a new feature for Android 9.

Battery saver improvements (#battery-saver)

When battery saver is turned on, the system places restrictions on all apps. This is an existing feature that is improved with Android 9.

Note: These changes apply to all apps, whether or not they target Android 9.

App Standby Buckets

Android 9 introduces a new battery management feature, App Standby Buckets. App Standby Buckets helps the system prioritize apps' requests for resources based on how recently and how frequently the apps are used. Based on the app usage patterns, each app is placed in one of five priority buckets. The system limits the device resources available to each app based on which bucket the app is in.

The five buckets prioritize apps into groups by the following characteristics:

Active

An app is in the active bucket if the user is currently using the app, for example:

- The app has launched an activity
- The app is running a foreground service
- The app has a sync adapter associated with a content provider used by a foreground app
- The user clicks on a notification from the app

If an app is in the active bucket, the system does not place any restrictions on the app's jobs, alarms, or FCM messages.

Working set

An app is in the working set bucket if it runs often but it is not currently active. For example, a social media app that the user launches most days is likely to be in the working set. Apps are also promoted to the working set bucket if they're used indirectly.

If an app is in the working set, the system imposes mild restrictions on its ability to run jobs and trigger alarms. For details, see Power management restrictions (/topic/performance/power/power-details).

Frequent

An app is in the frequent bucket if it is used regularly, but not necessarily every day. For example, a workout-tracking app that the user runs at the gym might be in the frequent bucket.

If an app is in the frequent bucket, the system imposes stronger restrictions on its ability to run jobs and trigger alarms, and also imposes a cap on high-priority FCM messages. For details, see Power management restrictions (/topic/performance/power/power-details).

Rare

An app is in the rare bucket if it is not often used. For example, a hotel app that the user only runs while they're staying at that hotel might be in the rare bucket.

If an app is in the rare bucket, the system imposes strict restrictions on its ability to run jobs, trigger alarms, and receive high-priority FCM messages. The system also limits the app's ability to connect to the internet. For details, see Power management restrictions (/topic/performance/power/power-details).

Never

Apps that have been installed but never run are assigned to the never bucket. The system imposes severe restrictions on these apps.

The system dynamically assigns each app to a priority bucket, and reassigns the apps as needed. The system may rely on a preloaded app that uses machine learning to determine how likely each app is to be used, and assigns apps to the appropriate buckets. If the system app is not present on a device, the system defaults to sorting apps based on how recently they were used. More active apps are assigned to buckets that give the apps higher priority, making more system resources available to the app. In particular, the bucket determines how frequently the app's jobs run, how often the app can trigger alarms, and how often the app can receive high-priority Firebase Cloud Messaging *(FCM)* (https://firebase.google.com/docs/cloud-messaging/) messages. These restrictions apply only while the device is on battery power; the system does not impose these restrictions on apps while the device is charging.

Every manufacturer can set their own criteria for how non-active apps are assigned to buckets. You should not try to influence which bucket your app is assigned to. Instead, focus on making sure your app behaves well in whatever bucket it might be in. Your app can find out what bucket it's currently in by calling the new method `UsageStatsManager.getAppStandbyBucket()` (/reference/android/app/usage/UsageStatsManager#getAppStandbyBucket()).

Note: Apps that are on the Doze allowlist (/training/monitoring-device-state/doze-standby#support_for_other_use_cases) are exempted from the app standby bucket-based restrictions.

Best practices

If your app is already following best practices for Doze and app standby (/training/monitoring-device-state/doze-standby), handling the new power management features should not be difficult. However, some app behaviors which previously worked well might now cause problems.

- Do not try to manipulate the system into putting your app into one bucket or another. The system's bucketing methods can change, and every device manufacturer could choose to write their own bucketing app with its own algorithm. Instead, make sure your app behaves appropriately no matter which bucket it's in.

- If an app does not have a launcher activity, it might never be promoted to the active bucket. You might want to redesign your app to have such an activity.
- If the app's notifications aren't actionable, users won't be able to trigger the app's promotion to the active bucket by interacting with the notifications. In this case, you may want to redesign some appropriate notifications so they allow a response from the user. For some guidelines, see the Material Design Notifications design patterns (https://material.io/guidelines/patterns/notifications.html).
- Similarly, if the app doesn't show a notification upon receiving a high-priority FCM message, it won't give the user a chance to interact with the app and thus promote it to the active bucket. In fact, the only intended use for high-priority FCM messages is to push a notification to the user, so this situation should never occur. If you inappropriately mark an FCM message as high-priority when it doesn't trigger user interaction, it can cause other negative consequences; for example, it can result in your app exhausting its quota, causing genuinely urgent FCM messages to be treated as normal-priority.

★ **Note:** If the user repeatedly dismisses a notification, the system gives the user the option of blocking that notification in the future. Do not spam the user with notifications just to try to keep your app in the active bucket!

- If apps are split across multiple packages, those packages might be in different buckets and, thus, have different access levels. You should be sure to test such apps with the packages assigned to various buckets to make sure the app behaves properly.

Battery saver improvements

Android 9 makes a number of improvements to battery saver mode. The device manufacturer determines the precise restrictions imposed. For example, on AOSP builds, the system applies the following restrictions:

- The system puts apps in app standby mode more aggressively, instead of waiting for the app to be idle.
- Background execution limits apply to all apps, regardless of their target API level.
- Location services may be disabled when the screen is off.
- Background apps do not have network access.

In addition, there are other, device-specific power optimizations. For full details, see the page that describes power management restrictions (/topic/performance/power/power-details).

As always, it's a good idea to test your app while battery saver is active. You can turn on battery saver manually through the device's **Settings > Battery Saver** screen.

Testing and troubleshooting

The new power management features affect all apps running on Android 9 devices, whether or not the apps target Android 9. It's important to make sure your app behaves properly on these devices.

Be sure to test your app's main use cases under a variety of conditions, to see how the power management features interact with each other. You can use Android Debug Bridge (/studio/command-line/adb) commands to turn some of the features on and off.

Android Debug Bridge commands

You can use Android Debug Bridge (/studio/command-line/adb) shell commands to test several of the power management features.

For information on using ADB to put your device in Doze, see Testing with Doze and App Standby (/training/monitoring-device-state/doze-standby).

App Standby Buckets

You can use ADB to manually assign your app to an App Standby Bucket. To change an app's bucket, use the following command:

```
$ adb shell am set-standby-bucket packagename active|working_set|frequent|rare
```

You can also use that command to set multiple packages at once:

```
$ adb shell am set-standby-bucket package1 bucket1 package2 bucket2...
```

To check what bucket an app is in, run

```
$ adb shell am get-standby-bucket [packagename]
```

If you don't pass a ***packagename*** parameter, the command lists the buckets for all apps. An app can also find out its bucket at runtime by calling the new method `UsageStatsManager.getAppStandbyBucket()` (/reference/android/app/usage/UsageStatsManager#getAppStandbyBucket()).

Battery saver

There are several commands to test how your app behaves in low-power conditions.

Note: You can also use the device **Settings > Battery saver** screen to put the device in battery saver mode.

To simulate the device being unplugged, use the command

```
$ adb shell dumpsys battery unplug
```

To test how the device behaves under low power conditions, use this command:

```
$ adb shell settings put global low_power 1
```

Once you have finished your testing, you can undo your manual device settings with this command:

```
$ adb shell dumpsys battery reset
```